IN THE CLAIMS

1. (Currently Amended) A method for handling exceptions in a multi-processor system, the method comprising:

receiving an exception within a processor which is one of a plurality of processors of the multi-processor system which is implemented within a control card of a network element for routing data in networks, wherein each of the plurality of processors in the multi-processor system shares a memory within the multi-processor system, wherein the memory includes a common interrupt handling vector address space shared by the plurality of the processors and a dedicated interrupt handling vector address space for each of the plurality of the processors; and

executing one or more instructions at an address associated with a type of the received exception within the common interrupt handling vector address space of the memory, wherein the one or more instructions cause the processor to modify based on an identification of the processor an execution flow of the received exception to execute an interrupt handler located within a respective dedicated interrupt handling vector address spaces associated with the processor,

wherein the plurality of processors includes a first processor and a second processor, the first processor executing a first operating system and the second processor executing a second operating system, the second operating system being different from the first operating system, and

wherein the first processor along with the first operating system is configured to handle routing of data received within the network element and the second processor

along with the second operating system is configured to handle provisioning and configuration of the network element.

2. (Currently Amended) The method of claim 1, wherein the first operating system is a real-time operating system handling the routing of data within the network element and the second operating system is a non real-time operating system handling the provisioning and configuration of the network element ~~each processor in the multi-processor system executes one of a plurality of operating systems, and wherein at least two of the operating systems are different~~.

3. (Currently Amended) The method of claim 2, wherein ~~each of~~ the first operating systems is associated with a first dedicated interrupt handling vector address spaces associated with ~~a processor executing the respective operating system~~the first processor, and wherein the second operating system is associated with a second dedicated interrupt handling vector address space associated with the second processor.

4. (Currently Amended) The method of claim 1, further comprising determining the identification of the first processor by reading a bit of a register within the first processor without having to access the memory.

5. (Currently Amended) The method of claim 4, wherein the register is part of a cache throttling register of the first processor, and wherein a dedicated bit of the register is used to indicate the identification of the first processor while at least a portion of remaining bits is used

for cache throttling purposes.

6. (Currently Amended) The method of claim 4, wherein determining the identification of the first processor comprises:

communicating during an initialization of the first processor with a memory controller coupling the first and second processors and the memory to retrieve the identification of the first processor from the memory controller; and

storing the retrieved identification of the first processor in the register within the first processor.

7. (Currently Amended) The method of claim 1, wherein each entry of the common interrupt handling vector space is associated with a different type of exceptions of the multi-processor system, and wherein a result of executing the instructions in the common interrupt handling vector space and the identification of the first processor receiving the exception determine the dedicated interrupt handling vector space associated with the first processor.

8. (Currently Amended) The method of claim 7, further comprising:

determining a type of the exception received by the first processor;

determining an entry of the common interrupt handling vector space based on the determined type of the exception; and

executing the one or more instructions stored at the determined entry to determine the dedicated interrupt handling vector space associated with the first processor based

on the identification of the first processor.

9. (Currently Amended) The method of claim 8, further comprising executing an interrupt service routine based on the determined entry of the dedicated interrupt handling vector space of the first processor to handle the exception.

10. (Currently Amended) The method of claim 6, wherein the memory controller includes a plurality of interfaces and each of the interfaces is coupled to one of the plurality of ~~processor~~processors, and wherein when each of the processors is being initialized during an initialization period of the multi-processor system, each of the plurality of the processors transmits a signal via a respective interface to the memory controller to identify ~~the respective~~each processor.

11. (Currently Amended) The method of claim 10, wherein in response to the signal received from each processor, the memory controller stores an identification of ~~a respective~~each processor within the memory controller, such that the identification of ~~the respective~~each processor is queried from the memory controller during runtime without having to access the memory.

12. (Currently Amended) A system comprising:

a plurality of processors including a first processor and a second ~~processors~~processor implemented within a control card of a network element for routing data for networks;

a memory that includes

a common exception handling vector address space shared by the plurality of processors, and

a plurality of exception handling vector address spaces each associated with each of the plurality of processors, including a first exception handling vector address space and a second exception handling vector address ~~spaces~~ space associated with the first processor and second ~~processors~~ processor respectively;

a memory controller coupled to the memory and the plurality of processors, wherein the first processor is to execute a first operating system, ~~and~~

wherein the second processor is to execute a second operating system, the second processor to execute one or more instructions in the common exception handling vector address space upon receipt of an exception,

wherein the one or more instructions cause the second processor to modify based on an identification of the second processor an execution flow of the exception to execute an interrupt handler located within a respective dedicated interrupt handling vector address spaces associated with the second processor,

wherein the the second operating system is different from the first operating system, and

wherein the first processor along with the first operating system is configured to handle routing of data received within the network element and the second processor along with the second operating system is configured to handle provisioning and configuration of the network element.

13. (Currently Amended) The system of claim 12, wherein the first ~~and second~~ operating system is a real-time operating system handling the routing of data within the network element and the second operating system is a non real-time operating system handling the provisioning and configuration of the network element ~~are different~~.

14. (Previously Presented) The system of claim 12, wherein the second processor further comprises an internal register to determine the identification of the second processor by reading a bit within the internal register.

15. (Previously Presented) The system of claim 14, wherein the internal register is part of a cache throttling register of the second processor, and wherein a dedicated bit of the internal register is used to indicate the identification of the second processor while at least a portion of remaining bits are used for cache throttling purposes.

16. (Previously Presented) The system of claim 14, wherein the identification of the second processor is determined by

communicating during an initialization of the second processor with the memory controller to retrieve the identification of the second processor from the memory controller, and

storing the retrieved identification of the second processor in the internal register within the second processor.

17. – 20. (Canceled)

21. (Currently Amended) A machine-readable medium that provides instructions for handling exceptions within a multi-processor system, which when executed by a machine, causes the machine to perform operations comprising:

receiving an exception within a processor which is one of a plurality of processors of the multi-processor system which is implemented within a control card of a network element for routing data in networks, wherein each of the plurality of processors in the multi-processor system shares a memory within the multi-processor system, wherein the memory includes a common interrupt handling vector address space shared by the plurality of the processors and a dedicated interrupt handling vector address space for each of the plurality of the processors; and

executing one or more instructions at an address associated with a type of the received exception within the common interrupt handling vector address space of the memory, wherein the one or more instructions cause the processor to modify based on an identification of the processor an execution flow of the received exception to execute an interrupt handler located within a respective dedicated interrupt handling vector address spaces associated with the processor,

wherein the plurality of processors includes a first processor and a second processor, the first processor executing a first operating system and the second processor executing a second operating system, the second operating system being different

from the first operating system, and

wherein the first processor along with the first operating system is configured to handle routing of data received within the network element and the second processor along with the second operating system is configured to handle provisioning and configuration of the network element.

22. (Currently Amended) The machine-readable medium of claim 21, wherein the first operating system is a real-time operating system handling the routing of data within the network element and the second operating system is a non real-time operating system handling the provisioning and configuration of the network element ~~each processor in the multi-processor system executes one of a plurality of operating systems, and wherein at least two of the operating systems are different~~.

23. (Previously Presented) The machine-readable medium of claim 22, wherein each of the operating systems is associated with a dedicated interrupt handling vector address spaces associated with a processor executing the respective operating system.

24. (Previously Presented) The machine-readable medium of claim 21, further comprising determining the identification of the processor by reading a bit of a register within the processor without having to access the memory.

25. (Previously Presented) The machine-readable medium of claim 24, wherein the register is

part of a cache throttling register of the processor, and wherein a dedicated bit of the register is used to indicate the identification of the processor while at least a portion of remaining bits of the register is used for cache throttling purposes.

26. (Previously Presented) The machine-readable medium of claim 21, wherein determining the identification of the processor comprises:

communicating during an initialization of the processor with a memory controller coupling the processors and the memory to retrieve the identification of the processor from the memory controller; and

storing the retrieved identification of the processor in the register within the processor.

27. (Previously Presented) The machine-readable medium of claim 21, wherein each entry of the common interrupt handling vector space is associated with a different type of exceptions of the multi-processor system, and wherein a result of executing the instructions in the common interrupt handling vector space and the identification of the processor receiving the exception determine the dedicated interrupt handling vector space associated with the processor.

28. – 39. (Canceled)